AD-A049 971 TEXAS UNIV AT AUSTIN CENTER FOR NUMERICAL ANALYSIS F/G 12/1
A ROAD MAP OF METHODS FOR APPROXIMATING SOLUTIONS OF TWO-POINT --ETC(U)
JAN 78 J W DANIEL N00014-76-C-0275
UNCLASSIFIED CNA-130 NL

END
DATE FILMED
3-78
DDC

12

# A ROAD MAP OF METHODS FOR APPROXIMATING SOLUTIONS OF TWO-POINT BOUNDARY-VALUE PROBLEMS*

by

James W. Daniel**

January 1978 CNA-130

*To be presented by invitation at the Working Conference on Codes for Boundary-value Problems in ODEs in Houston, May 1978.

**Departments of Mathematics and of Computer Science, and Center for Numerical Analysis, The University of Texas, Austin, Texas.



CENTER FOR NUMERICAL ANALYSIS

THE UNIVERSITY OF TEXAS AT AUSTIN

# A ROAD MAP OF METHODS FOR APPROXIMATING SOLUTIONS OF TWO-POINT BOUNDARY-VALUE PROBLEMS



by

James W. Daniel*

Abstract

Numerical methods for approximating solutions of two-point boundary-value problems for ordinary differential equations are surveyed. Specific complete algorithms are classified according to how the original problem is transformed, how the transformed problem is modeled discretely, and how the discrete model is solved. Relationships among various complete algorithms are presented. Convergence acceleration, error estimation and control, and parameter selection are also discussed.

Key Words: Boundary-value, ordinary differential equations, numerical methods.

**********

## 1. INTRODUCTION

I intentionally avoided calling this paper a "survey" because, having once worked as a surveyor, I know that a survey of a city gives an extremely detailed description of the precise layout of the property in that city

* Departments of Mathematics and of Computer Sciences and Center for Numerical Analysis at The University of Texas at Austin. Research supported in part by the United States Office of Naval Research under Contract N00014-76-C-0275; reproduction in whole or in part is permitted for any purposes of the United States government.

and is not very helpful to someone trying to find his or her way around town. Analogously, presenting all the details of various implemented methods for solving boundary-value problems can obscure the concepts. It is also true, however, that a coarse aerial photograph of a city is a poor guide for the lost traveler, and, analogously, a very abstract model representing all methods for boundary-value problems is too general to impart much information. What both the traveler and the student of numerical methods need is a useful roadmap with not only enough detail to show the various points of interest but also enough perspective to show where these sites lie in relation to one another. Here I present my own such roadmap of what some numerical methods for boundary-value problems <u>are</u>, of how they relate to one another, and of what areas need development in order to improve methods. (My use of "I" as the first word of this paper is also intentional: the reader is to be warned that this is the personal view of one individual.)

Now, what kinds of boundary-value problems are we to consider? In the spirit of the first paragraph I want to present neither a single abstract problem including all cases nor a vast list of specific special problems. I will discuss instead a couple of model problems for which the solution methods will share many features with methods for the panorama of distinct problem types: eigenvalues, non-linear boundary conditions, m-th order equations, systems of equations for vector-valued functions, mixed-order systems, infinite-intervals for the independent variable, singular problems, singular-perturbation problems, multi-point boundary conditions, et cetera. I will consider both the <u>first-order system</u> for the n x 1 vector valued function $\underline{y}$:

(1.1) $\underline{y}'(t) = \underline{f}(t,\underline{y}(t))$ for $0 < t < 1$

and the second-order scalar equation:

(1.2) $y''(t) = f(t,y(t),y'(t))$ for $0 < t < 1$

since numerical methods on the first-order system equivalent to (1.2) usually are dramatically less efficient than methods directly intended for second-order problems; note that I restrict myself to a finite range for the independent variable and I use $0 < t < 1$ as a canonical interval. Boundary conditions for (1.1) are given by $n$ nonlinear equations involving $\underline{y}(0)$ and $\underline{y}(1)$:

(1.3) $\underline{b}(\underline{y}(0), \underline{y}(1)) = \underline{0}$

in vector notation. For (1.2) we give two nonlinear equations relating $y(0)$, $y'(0)$, $y(1)$, and $y'(1)$, which we can express in vector notation:

(1.4) $\underline{c}(y(0), y'(0), y(1), y'(1)) = \underline{0}$ .

In many cases the boundary conditions will in fact be linear, in which case we replace (1.3) with

(1.5) $\underline{\underline{B}}_0\underline{y}(0) + \underline{\underline{B}}_1\underline{y}(1) = \underline{e}$

where $\underline{\underline{B}}_0$ and $\underline{\underline{B}}_1$ are $n \times n$ and $\underline{e}$ is $n \times 1$, while we replace (1.4) with

(1.6) $\underline{c}_0 y(0) + \underline{d}_0 y'(0) + \underline{c}_1 y(1) + \underline{d}_1 y'(1) = \underline{a}$

where $\underline{c}_0$, $\underline{d}_0$, $\underline{c}_1$, $\underline{d}_1$, and $\underline{a}$ are all $2 \times 1$; these special forms can be useful computationally. Another common and computationally advantageous situation is that in which the boundary conditions are <u>separated</u>, so that conditions at $t = 0$ and at $t = 1$ do not interact. In this case we can write (1.3) and (1.5) as

$$\tilde{B}_0 \underline{y}(0) = \underline{e}_0, \quad \tilde{B}_1 \underline{y}(1) = \underline{e}_1 \tag{1.7}$$

where $\tilde{B}_0$ is $q \times n$, $\tilde{B}_1$ is $(n-q) \times n$, $\underline{e}_0$ is $q \times 1$, and $\underline{e}_1$ is $(n-q) \times 1$, for some integer $q$ with $1 < q < n$. Similarly (1.4) and (1.6) become in the separated case

$$c_0 y(0) + d_0 y'(0) = a_0, \quad c_1 y(1) + d_1 y'(1) = a_1. \tag{1.8}$$

Thus we will be considering either (1.1) with one of the boundary conditions (1.3), (1.5), (1.7) or (1.2) with one of the boundary conditions (1.4), (1.6), (1.8). In the interest of time and space we often will discuss a method as applied to <u>either</u> the first-order <u>or</u> the second-order problem when the analogous use of the idea of the method for the other standard problem is fairly straightforward.

The next task and the main task of this paper is to describe how to classify various methods. The "aerial photograph" approach would be to note that the problem is simply to solve $\underline{F}(\underline{x}) = \underline{0}$ for $\underline{x}$ in some appropriate abstract space and $\underline{F}$ some nonlinear operator, while numerical methods eventually solve some discretization $\underline{F}_h(\underline{x}_h) = \underline{0}_h$ for $\underline{x}_h$ in some discretized (finite-dimensional) space; this doesn't really tell us much about the structure of various specific methods. The "survey" approach would be to describe computer codes implementing various specific

methods; this gives us more detail than we can absorb. Instead I will give a "road map" approach which defines a complete method as having three aspects:

1) a Transformed Problem,

2) a Discrete Model of the Transformed Problem, and

3) a Solution Technique for the Discrete Model.

In Section 2 of this paper I describe various Transformed Problems equivalent to (1.1) or (1.2) and their boundary conditions. Section 3 presents approaches for creating Discrete Models, while Section 4 outlines some Solution Techniques. In Section 5 complete methods generated by different choices of 1), 2), and 3) above are compared and some are shown to be equivalent to others. Sections 6 and 7 briefly discuss the important notions of accelerating convergence and estimating errors, and of controlling errors by selecting the parameters upon which various complete methods depend. A brief conclusion appears in Section 8. Section 9 contains a brief collection of references. Since I am not attempting here to give a historical or developmental view of methods, I will not present a detailed bibliography but rather will indicate selected references where more detail and more references may be found. Good general sources of references are [Keller (1968, 1975), Aktas-Stetter (1977) ].

I thank the many colleagues, especially Victor Pereyra and Andy White, who over the years have influenced my view of numerical methods for boundary-value problems. Although I would gladly blame mistakes on my colleagues and take credit for any insights, unfortunately I must accept responsibility for all the views expressed in this paper.

## 2. TRANSFORMED PROBLEMS

The basic notion of this paper is that a complete method can be viewed as the straightforward application of a fairly standard discretization process to a Transformed Problem that is equivalent to the original boundary-value problem. From this perspective we will describe the classical shooting method, for example, as the application of standard numerical methods for initial-value problems and for nonlinear equations to the Transformed Problem of finding the proper initial values so as to satisfy the boundary conditions. In this Section we examine several Transformed Problems.

2.1 No transformation. For completeness we mention the trivial case of applying no transformation, so that the Transformed Problem and the original problem coincide. This allows us to discuss complete methods which appear to be frontal assaults on the boundary-value problem.

2.2 Variational problems. Many boundary-value problems arise in the physical sciences as the variational or Euler-Lagrange equations for problems in the calculus of variations [Courant-Hilbert (1953)]. For example, consider the problem of minimizing

$$J[y] = \int_0^1 \{ \frac{1}{2} (y'(t))^2 + F(t,y(t)) \} dt \tag{2.1}$$

for all sufficiently smooth functions $y$ satisfying the linear and separated boundary conditions

$$y(0) = a, \quad y(1) = b \ . \tag{2.2}$$

Then the theory of the calculus of variations shows that

(2.3) $\qquad y''(t) = f(t,y(t)) \quad \text{for } 0 < t < 1,\ y(0) = a,\ y(1) = b$

where $f(t,y) = \frac{\partial}{\partial y} F(t,y)$. Thus the calculus of variations problem of minimizing $J$ in (2.1) reduces to a special form of the boundary-value problem (1.2) in which no $y'$ term appears. Conversely, certain boundary-value problems (2.3)--for example, those in which $\frac{\partial f}{\partial y} \geq 0$ for all $t$ and $y$--are equivalent to minimizing $J[y]$ and are perhaps more naturally described as such variational problems. In this case, we change the original boundary-value problem to the Transformed Problem: find a function $y$ minimizing $J[y]$ in (2.1) subject to conditions (2.2).

2.3 Shooting and its variants. We consider first here the simplest form of shooting applied to the first-order system (1.1) with boundary conditions (1.3), that is

$$\underline{y}' = \underline{f}(t,\underline{y}), \quad \text{for } 0 < t < 1,\ \underline{b}(\underline{y}(0),\underline{y}(1)) = \underline{0} \ .$$

We choose an initial-value vector $\underline{z}$ that is $n \times 1$ and let $\underline{y}(t;\underline{z})$ solve (1.1) subject to the initial condition

$$\underline{y}(0;\underline{z}) = \underline{z} \ .$$

The original problem now can be restated as the Transformed Problem: find an $n \times 1$ vector $\underline{z}$ so that $\underline{b}(\underline{z},\underline{y}(1;\underline{z})) = \underline{0}$. We have replaced a boundary-value problem for differential equations with a single system of nonlinear equations (and an intermediate problem of finding $\underline{y}(1;\underline{z})$). It is easy to see one of the potential difficulties in using simple shooting (independent of the numerical technique used to compute $\underline{y}(1;\underline{z})$) by examining a variant of shooting called superposition [Scott-Watts (1977)].

Suppose that our first-order system (1.1) is linear with linear boundary conditions (1.5) so that

$$\underline{y}'(t) = \underline{A}(t)\underline{y}(t) + \underline{g}(t) \quad \text{for} \quad 0 < t < 1, \ \underline{B}_0\underline{y}(0) + \underline{B}_1\underline{y}(1) = \underline{e} \tag{2.4}$$

where $\underline{A}$ is n x n. We let $\underline{Y}(t)$ be the n x n fundamental solution matrix satisfying

$$\underline{Y}'(t) = \underline{A}(t)\underline{Y}(t) \quad \text{for} \quad 0 < t < 1, \ \underline{Y}(0) = \underline{I}$$

and let the n x 1 vector $\underline{p}$ be any particular solution to

$$\underline{p}'(t) = \underline{A}(t)\underline{p}(t) + \underline{g}(t) \quad \text{for} \quad 0 < t < 1 \ .$$

Then every solution $\underline{y}$ to (2.4) is of the form

$$\underline{y}(t) = \underline{p}(t) + \underline{Y}(t)\underline{x} \quad \text{for } 0 < t < 1 \tag{2.5}$$

for some n x 1 vector $\underline{x}$ independent of t; this representation merely writes $\underline{y}$ as $\underline{p}$ plus a linear combination of a linearly independent set of solutions to the homogeneous equations. The boundary conditions now merely become the linear algebraic equations for $\underline{x}$:

$$[\underline{B}_0 + \underline{B}_1\underline{Y}(1)]\underline{x} = \underline{e} - \underline{B}_0\underline{p}(0) - \underline{B}_1\underline{p}(1) \ .$$

To determine $\underline{x}$ it is essential of course that $\underline{B}_0 + \underline{B}_1\underline{Y}(1)$ be non-singular; the difficulty can be that the numerical computation of $\underline{Y}$ causes the crucial matrix to become singular. For example, while

$\begin{bmatrix} e^{10t} & e^{10t}-e^{-10t} \\ e^{10t}-e^{-10t} & e^{10t} \end{bmatrix}$ is non-singular for every $t$, because of rounding errors a computer representation of this matrix will become the singular matrix $\begin{bmatrix} e^{10t} & e^{10t} \\ e^{10t} & e^{10t} \end{bmatrix}$ for $t$ much larger than unity. Thus simple superposition can have difficulties; since from (2.5) we see that $\underline{y}(0) = \underline{p}(0) + \underline{x}$, the coefficients $\underline{x}$ are nearly the initial values $\underline{z}$ of shooting, and indeed $\underline{x} = \underline{z}$ if $\underline{p}(0) = \underline{0}$, so we see that the same potential difficulty is inherent in shooting.

From the shooting viewpoint, a way out of this problem is <u>multiple shooting</u> [Keller (1968] in which we we simultaneously shoot from $k$ distinct $t$ values $0 = T_1 < \cdots < T_k < 1$. That is, for arbitrary $n \times 1$ vectors $\underline{z}_1, \cdots, \underline{z}_k$ we let $\underline{y}_i(t;\underline{z}_i)$ for $1 \le i \le k$ solve $\underline{y}_i'(t) = \underline{f}(t, \underline{y}_i(t))$ for $T_i < t < T_{i+1}$ (with $T_{k+1} \equiv 1$) subject to the initial conditions $\underline{y}_i(T_i;\underline{z}_i) = \underline{z}_i$. The original problem now can be restated as the Transformed Problem: find $k$ $n \times 1$ vectors $\underline{z}_1, \cdots, \underline{z}_k$ so that $\underline{b}(\underline{z}_1, \underline{y}_k(1;\underline{z}_k)) = 0$, $\underline{y}_i(T_{i+1};\underline{z}_i) = \underline{z}_{i+1}$ for $1 \le i \le k-1$, a system of $kn$ equations for the $kn$ unknown components of $\underline{z}_1, \ldots, \underline{z}_k$. The hope is, of course, that the $\underline{z}_i$ and $T_i$ can be chosen so shrewdly that the numerically singular matrices mentioned in the preceding paragraph do not arise.

From the superposition viewpoint we can use multiple starting points $T_1 = 0 < T_2 < \cdots < T_k < 1$ as well. We merely represent $\underline{y}$ for $T_i \le t \le T_{i+1}$ by $\underline{y}(t) = \underline{Y}_i(t)\underline{x}_i + \underline{p}_i(t)$ where $\underline{p}_i$ is a particular solution of the inhomogeneous equation on $T_i < t < T_{i+1}$ and $\underline{Y}_i$ is a fundamental solution there solving $\underline{Y}_i' = \underline{A}(t)\underline{Y}_i$ with $\underline{Y}_i(T_i) = \underline{Y}_{i,0}$ for some given non-singular

matrix $\underline{\underline{Y}}_{i,0}$. Recalling that we need to satisfy the linear boundary conditions $\underline{\underline{B}}_0\underline{y}(0) + \underline{\underline{B}}_1\underline{y}(1) = \underline{e}$ and the continuity conditions for $\underline{y}(T_i)$ which become

$$\underline{p}_{i-1}(T_i) + \underline{\underline{Y}}_{i-1}(T_i)\underline{x}_{i-1} = \underline{p}_i(T_i) + \underline{\underline{Y}}_{i,0}\underline{x}_i \quad \text{for} \quad 2 \le i \le k \,, \tag{2.7}$$

we see that we again have $k\,n$ (linear) equations for the $k\,n$ unknown components of $\underline{x}_1, \cdots, \underline{x}_k$. Since $\underline{x}_{i-1}$ and $\underline{x}_i$ are the unknowns in (2.7) we see that the system of linear equations we must solve is essentially block bi-diagonal with $\underline{\underline{Y}}_{i-1}(T_i)$ and $\underline{\underline{Y}}_{i,0}$ appearing as the blocks. Just as for simple shooting and simple superposition, if we choose $\underline{\underline{Y}}_{i,0} = \underline{\underline{I}}$ and $\underline{p}_i(T_i) = \underline{0}$ then the coordinates $\underline{x}_i$ in multiple superposition equal the initial values $\underline{z}_i$ in multiple shooting. Other choices of $\underline{\underline{Y}}_{i,0}$ and $\underline{p}_i(T_i)$ are possible however. In the so-called <u>re-orthogonalization method</u> [Scott-Watts (1977)], $\underline{\underline{Y}}_{1,0} = \underline{\underline{I}}$ and each subsequent $\underline{\underline{Y}}_{i,0}$ is chosen as the Gram-Schmidt orthogonalized version of $\underline{\underline{Y}}_{i-1}(T_i)$. Thus we decompose $\underline{\underline{Y}}_{i-1}(T_i)$ as

$$\underline{\underline{Y}}_{i-1}(T_i) = \underline{\underline{Q}}_i\underline{\underline{R}}_i$$

for orthogonal $\underline{\underline{Q}}_i$ and upper-right triangular $\underline{\underline{R}}_i$ and then let $\underline{\underline{Y}}_i(T_i) = \underline{\underline{Y}}_{i,0} \equiv \underline{\underline{Q}}_i$. In this case the equations (2.7) for the coefficients $\underline{x}_1, \cdots, \underline{x}_k$ become

$$\underline{p}_{i-1}(T_i) + \underline{\underline{Q}}_i\underline{\underline{R}}_i\underline{x}_{i-1} = \underline{0} + \underline{\underline{Q}}_i\underline{x}_i \,,$$

and multiplying by the inverse $\underline{\underline{Q}}_i^T$ of the orthogonal matrix $\underline{\underline{Q}}_i$ gives

$$\underline{\underline{Q}}_i^T\underline{p}_{i-1}(T_i) + \underline{\underline{R}}_i\underline{x}_{i-1} = \underline{x}_i \,. \tag{2.8}$$

Thus in the case of the re-orthogonalization method the nearly block bi-diagonal matrix describing the equations for $\underline{x}_1,\cdots,\underline{x}_k$ has the very simple blocks $\underline{I}$ and $\underline{R}_i$, making solution of the system quite simple.

Whatever variants we take of simple shooting, they all reduce the original boundary-value problem to a Transformed Problem of determining a finite set of numbers--$\underline{z}$ or $\underline{x}$ or $\underline{z}_1,\cdots,\underline{z}_k$ or $\underline{x}_1,\cdots,\underline{x}_k$.

2.4 Quasi-linearization. Mathematicians are well known as people who, when they cannot solve a certain difficult problem, instead solve some easy problem in the hope that this will somehow be profitable. Quasi-linearization [Bellman-Kalaba (1965)], often known as Newton's method, is the application of this ploy to create Transformed Problems simpler than an original difficult nonlinear boundary-value problem. The idea is that, given one approximate solution $\underline{y}_i(t)$ to the first-order system (1.1), with general boundary conditions (1.3), we approximate the differential equation (1.1) for $\underline{y}$ near $\underline{y}_i$ by the linear (in $\underline{y}_{i+1}$) differential equation

$$\underline{y}'_{i+1}(t) = \underline{f}(t,\underline{y}_i(t)) + \frac{\partial \underline{f}}{\partial \underline{y}}(t,\underline{y}_i(t))(\underline{y}_{i+1}(t)-\underline{y}_i(t)), \tag{2.9}$$

where $\frac{\partial \underline{f}}{\partial \underline{y}}$ is the n x n Jacobian matrix of $\underline{f}$ with respect to $\underline{y}$. Similarly we approximate the boundary conditions (1.3) by the linear (in $\underline{y}_{i+1}$) boundary conditions

$$\begin{cases} \underline{b}(\underline{y}_i(0),\underline{y}_i(1)) + \underline{B}_{0,i}(\underline{y}_{i+1}(0) - \underline{y}_i(0)) + \underline{B}_{1,i}(\underline{y}_{i+1}(1) - \underline{y}_i(1)) = \underline{0} \\ \underline{B}_{0,i} = \frac{\partial \underline{b}}{\partial \underline{y}(0)}\,\underline{b}(\underline{y}_i(0),\underline{y}_i(1)), \\ \underline{B}_{1,i} = \frac{\partial \underline{b}}{\partial \underline{y}(1)}\,\underline{b}(\underline{y}_i(0),\underline{y}_i(1)) \end{cases} \tag{2.10}$$

The equations (2.9), (2.10) comprise a linear boundary-value problem for $\underline{y}_{i+1}$ of the same form as (2.4). As usual with Newton's method, the hope

is that the sequence of functions $\underline{y}_0, \underline{y}_1, \ldots$ converges to $\underline{y}$ solving (1.1), (1.3); under reasonable conditions on $\underline{f}$ and $\underline{b}$ this does indeed occur if the initial guess $\underline{y}_0(t)$ is sufficiently close to $\underline{y}(t)$. Thus we have reduced the original nonlinear boundary-value problem to the Transformed Problem: solve a sequence of linear boundary-value problems for $\underline{y}_0, \underline{y}_1, \ldots$ converging to $\underline{y}$.

At this point notice that we can easily speak of Transformed$^2$ Problems whenever the Transformed Problem is transformed again; this occurs for example if we use shooting or superposition as in Subsection 2.3 to solve each of the linear boundary-value problems (2.10). Some methods designed only for linear problems can thus be used with quasilinearization on nonlinear problems [for example, see Scott (1975), Scott-Watts (1977)].

2.5 Continuation and embedding. In many real problems the differential equation (and perhaps the boundary conditions) depends on certain physical parameters, and solutions are desired over a range of values of these parameters. If this is not the case, it is usually possible to think of the problem as being for one specific value, say $\lambda_F$, of some physically meaningful parameter $\lambda$ which we can imagine being allowed to vary. In rare instances an artificial parameter may need to be introduced by, for example, considering $\underline{y}' = \underline{f}(t,\underline{y})$ as the problem when $\lambda = \lambda_F = 1$ is substituted in the equation $\underline{y}' = \lambda \underline{f}(t,\underline{y})$. In any case we suppose that we have a family of differential equations

$$\underline{y}' = \underline{f}(t,\underline{y};\lambda) \tag{2.11}$$

whose solution $\underline{y}(t;\lambda)$ is especially desired for $\lambda = \lambda_F$ and perhaps for many other values of $\lambda$ as well.

In the continuation method we assume that (2.11) can be solved easily for some value $\lambda_0$ of the parameter. Assuming $\lambda_0 < \lambda_F$ for convenience, we then set out to solve (2.11) for a sequence of $k$ values of $\lambda$, say $\lambda_1 = \lambda_0 < \lambda_2 < \cdots < \lambda_k = \lambda_F$. If $\lambda_{i+1}$ is "near" $\lambda_i$, the hope is that (2.11) for $\lambda = \lambda_{i+1}$ can be solved fairly easily by making use of the already obtained solution $\underline{y}(t;\lambda_i)$ at $\lambda = \lambda_i$. Thus we have replaced the original boundary-value problem (1.1) by the Transformed Problem: solve (2.11) for a sequence $\lambda_0, \lambda_1, \cdots, \lambda_k$ of values of the parameter $\lambda$. This is a useful device whenever one can use $\underline{y}_i$ to advantage in obtaining $\underline{y}_{i+1}$; since any numerical method will require solving some nonlinear equations for the approximation to $\underline{y}_{i+1}$, the approximation obtained for $\underline{y}_i$ can usually be used as the first iterate in an iterative method for solving these nonlinear equations for $\underline{y}_{i+1}$. For example, quasi-linearization might be used on (2.11) at $\lambda = \lambda_{i+1}$ with the solution at $\lambda = \lambda_i$ as the starting iterate.

Another approach to solving (2.11) for $\lambda = \lambda_F$ is to use the embedding method [Scott (1973,1975)] which derives differential equations for the dependence of $\underline{y}(t;\lambda)$ on $\lambda$. While this usually results in nonlinear partial differential equations for $\underline{y}$ as a function of $t$ and $\lambda$, the side conditions often can be chosen to be initial conditions since we assumed the problem to be easily solved initially at $\lambda = \lambda_0$. Thus the original ordinary differential equation boundary-value problem is replaced by the Transformed Problem: solve an initial-value problem for a partial differential equation involving $\underline{y}$ as a function of $t$ and $\lambda$.

A wide variety of these embedding methods have been used depending on precisely how $\lambda$ enters the differential equation. A very common practice is to use the interval length over which $t$ varies as the

embedding parameter; it is for this case that I will restrict the use of the broad term invariant embedding [Scott (1973,1975)]. Thus we think of the family of problems defined for $0 < t < \lambda$ and we let $\lambda$ range from zero to unity. For (1.1) subject to (1.3), for example, we instead consider

$$\underline{y}' = \underline{f}(t,\underline{y}) \quad \text{for } 0 < t < \lambda \text{ with } \underline{b}(\underline{y}(0),\underline{y}(\lambda)) = \underline{0}.$$

When $\lambda = \lambda_0 \equiv 0$ this reduces to the system of $n$ equations $\underline{b}(\underline{y}(0),\underline{y}(0)) = \underline{0}$ for the $n$ unknown components of $\underline{y}(0;\lambda_0)$ which is assumed to be our "easy" problem. When the original differential equation (or boundary condition) is nonlinear, invariant embedding yields a nonlinear partial differential equation. To avoid this difficulty the computationally most successful approach appears [Scott (1973, 1975)] to be to develop invariant embedding for linear problems (which turn out to lead to ordinary differential equations when invariant embedding is used) and then to use quasi-linearization as a device for replacing nonlinear problems by a sequence of linear problems, each of which is transformed and solved by invariant embedding; of course this can be viewed as just one possible device for solving the nonlinear partial differential equation. For the rest of this section we restrict ourselves to a consideration of the linear second-order equation

$$y''(t) + p(t)y'(t) + q(t)y(t) = g(t) \quad \text{for } 0 < t < \lambda \tag{2.12}$$

subject to separated linear boundary conditions

$$c_0 y(0) + d_0 y'(0) = a_0, \quad c_1 y(\lambda) + d_1 y'(\lambda) = a_1 \tag{2.13}$$

as in (1.8); the solution to (2.12), (2.13) is $y(t;\lambda)$ and it is desired

for $0 \le \lambda \le 1$. General linear boundary conditions can be handled as well.

The simplest invariant embedding method for (2.12), (2.13) is the sweep or factorization method in which we introduce two auxiliary functions $\alpha(t)$ and $\beta(t)$ for $0 \le t \le 1$ and set $y' = \alpha y + \beta$; it turns out that $\alpha$ and $\beta$ must satisfy

$$(2.14) \qquad \begin{cases} \alpha'(t) = -q(t) - p(t)\alpha(t) - \alpha^2(t) & \text{for } 0 < t < 1, \ \alpha(0) = \dfrac{-c_0}{d_0} \\ \beta'(t) = g(t) - (p(t) + \alpha(t))\beta(t) & \text{for } 0 < t < 1, \ \beta(0) = \dfrac{a_0}{d_0} \end{cases} .$$

(If $d_0 = 0$, a slightly different method is used.) Having computed $\alpha$ and $\beta$ from (2.14), applying $y' = \alpha y + \beta$ at $t = \lambda$ along with the boundary condition $c_1 y(\lambda) + d_1 y'(\lambda) = a_1$ gives us two linear equations which we solve for the unknowns $y(\lambda;\lambda)$ and $y'(\lambda;\lambda)$. Having found $y(\lambda;\lambda)$ and $\alpha(t), \beta(t)$ for $0 \le t \le \lambda \le 1$ we finally solve

$$(2.15) \qquad y'(t;\lambda) = \alpha(t)y(t;\lambda) + \beta(t) \quad \text{for } 0 < t < \lambda$$

in the backward direction starting from the recently found value $y(\lambda;\lambda)$ for $y(t;\lambda)$ at $t = \lambda$. This gives the desired solution $y(t;\lambda)$ for $0 \le t \le \lambda$. Note that the initial-value problems for $\alpha$ and $\beta$ need only be solved once. Thereafter, to find $y(t;\lambda)$ for any $\lambda$ only requires the solution of the two linear algebraic equations for $y(\lambda;\lambda)$ and then integration of one backwards initial-value problem (2.15) for $y$.

Experience has indicated [Scott (1975)] that a somewhat more complex invariant embedding method is better than the sweep method above. In this version four auxiliary functions $r_1$, $r_2$, $s_1$, and $s_2$ are introduced in such a fashion that

$$(2.16)\qquad \begin{cases} y(t;\lambda) = r_1(t)y'(t;\lambda) + r_2(t) \\ y'(0;\lambda) = s_1(t)y'(t;\lambda) + s_2(t) \end{cases} .$$

From (2.16) we see that <u>if</u> we know $r_1$, $r_2$, $s_1$, and $s_2$ for $0 \le t \le 1$ and if we know $y'(0;\lambda)$ then (2.16) gives

$$(2.17)\qquad y(t;\lambda) = r_1(t)\ \frac{y'(0;\lambda) - s_2(t)}{s_1(t)} + r_2(t)$$

which expresses $y(t;\lambda)$ in terms of known quantities. To determine $r_1$, $r_2$, $s_1$, $s_2$, and $y'(0;\lambda)$ we proceed as follows. We find $r_1$, $r_2$, $s_1$, $s_2$ for $0 \le t \le 1$ by solving the initial-value problems

$$(2.18)\qquad \begin{cases} r_1'(t) = 1 + p(t)r_1(t) + q(t)r_1^2(t), & r_1(0) = 0, \\ r_2'(t) = q(t)r_1(t)r_2(t), & r_2(0) = 1, \\ s_1'(t) = (p(t) + q(t)r_1(t))s_1(t), & s_1(0) = 1, \\ s_2'(t) = (-g(t) + q(t)r_2(t))s_1(t), & s_2(0) = 0. \end{cases}$$

To obtain $y'(0;\lambda)$ we use the two equations of (2.16) for $t = \lambda$ and we also use the two boundary conditions (2.13); this gives four linear algebraic equations from which we evaluate the four unknowns $y(0;\lambda)$, $y'(0;\lambda)$, $y(\lambda;\lambda)$, and $y'(\lambda;\lambda)$. Note again that the initial-value problem for $r_1$, $r_2$, $s_1$, and $s_2$ are solved only <u>once</u>; thereafter for any value of $\lambda$ we need only solve the four algebraic equations to find $y'(0;\lambda)$ and substitute into (2.17) to obtain the full solution $y(t;\lambda)$.

Thus for both the sweep method and Scott's version of invariant embedding, we replace the original boundary-value problem by the Transformed Problem: solve three or four initial-value problems (for ordinary differential equations) and one small linear system of algebraic equations.

2.6 Integral Equations. By using an appropriate Green's function we can transform our original boundary-value problem into an integral equation. As an illustration, consider the second-order problem (1.2) subject to linear boundary conditions. Usually by subtracting from $\underline{y}$ an appropriate linear function, we can force the boundary conditions to be homogeneous. We therefore consider

$$y''(t) = f(t,y(t),y'(t)) \quad \text{for} \quad 0 < t < 1 \tag{2.19}$$

subject to the homogeneous version of (1.6), namely

$$\underline{c}_0 y(0) + \underline{d}_0 y'(0) + \underline{c}_1 y(1) + \underline{d}_1 y'(1) = \underline{0} \tag{2.20}$$

If $y = 0$ is the only solution to $y'' = 0$ subject to (2.20) then we can find the Green's function $G(t,\tau)$ so that $y''(t) = g(t)$ and $y$ satisfies (2.20) if and only if

$$y(t) = \int_0^1 G(t,\tau)g(\tau)d\tau.$$

Applying this fact to $g(t) = f(t,y(t),y'(t))$ in (2.19) yields the fact that $y$ solves (2.19), (2.20) if and only if $y$ solves

$$y(t) = \int_0^1 G(t,\tau)f(\tau,y(\tau),y'(\tau))d\tau \quad \text{for } 0 \le t \le 1 \tag{2.21}$$

In this generality we have replaced (2.19), (2.20) by an integro-differential equation. For the important class of problems in which $f$ is independent of $y'$, (2.21) becomes the integral equation

$$(2.22) \qquad y(t) = \int_0^1 G(t,\tau)f(\tau,y(\tau))d\tau \quad \text{for } 0 \le t \le 1.$$

Thus we replace the original boundary-value problem by a Transformed Problem: solve for $y$ in the integro-differential equation (2.21) or in the integral equation (2.22)

## 3. DISCRETE MODELS OF TRANSFORMED PROBLEMS

We have seen a few of the many ways in which our original boundary-value problem can be transformed into an equivalent problem; now we want to discuss methods for launching a frontal assault on the Transformed Problem. Although there are other methods available [Aktas-Stetter (1977)], I will restrict myself to the most successful methods, namely finite differences and projections, as approaches to discrete modeling.

3.1 Finite Differences. The basic idea here is to represent desired functions $g(t)$ for $0 \le t \le 1$ by the values of $g$ at some finite set of points $0 \le t_1 < t_2 < \cdots < t_N \le 1$. We approximate $g(t_i)$ by some number $G_i$ and generate relationships among the values $G_i$ intended to model what the (transformed) problem tells us about $g$. Such modeling methods are, of course, very well known: we generally replace derivatives by divided differences, integrals by quadrature sums, et cetera. We look briefly at the models that result when finite differences are used with the Tranformed Problems of Section 2.

## Finite Differences for the Original Problem

Basic finite differences for the original boundary-value problem are, of course, very well known. For the first-order system (1.1), for example, we discretize by letting $t_1 = 0 < t_2 < \cdots < t_N = 1$ and letting the $n \times 1$ vector $\underline{Z}_i$ approximate $\underline{y}(t_i)$. Two simple, natural, and effective schemes are to model the differential equation (1.1) either by

$$(\underline{Z}_{i+1} - \underline{Z}_i)/(t_{i+1} - t_i) = \frac{1}{2}\underline{f}(t_i, \underline{Z}_i) + \frac{1}{2}\underline{f}(t_{i+1}, \underline{Z}_{i+1}) \quad , \tag{3.1}$$

$$\text{for } 1 \le i \le N-1$$

or by

$$(\underline{Z}_{i+1} - \underline{Z}_i)/(t_{i+1} - t_i) = \underline{f}\left(\frac{1}{2}t_i + \frac{1}{2}t_{i+1}, \frac{1}{2}\underline{Z}_i + \frac{1}{2}\underline{Z}_{i+1}\right) , \tag{3.2}$$

$$\text{for } 1 \le i \le N-1$$

and to model the nonlinear boundary conditions $\underline{b}(\underline{y}(0), \underline{y}(1)) = \underline{0}$ in (1.3) by

$$\underline{b}(\underline{Z}_1, \underline{Z}_N) = \underline{0} \quad . \tag{3.3}$$

Under reasonable hypotheses, of course, this is a second-order method, that is,

$$\|\underline{Z}_i - \underline{y}(t_i)\|_\infty \le ch^2 \quad \text{for } 1 \le i \le N, \text{ c independent of N}, \tag{3.4}$$

where throughout this paper we use $h$ to denote

$$h = \max\{|t_{i+1} - t_i|; \ 1 \le i \le N-1\}. \tag{3.5}$$

Higher order methods of course exist using more complicated difference approximations for $\underline{y}'$ and/or more complicated sums of values of $\underline{f}$. Such methods can even be generated automatically as in the HODIE method [Lynch-Rice (1977)].

Since the nonlinear equations (3.1), (3.3) or (3.2), (3.3) are often solved by some linearization process and, since linear problems also arise naturally, it is instructive to look briefly at the structure of, say, (3.2) and (3.3), when the problem is linear. We therefore consider again (2.4), namely

$$\underline{y}'(t) = \underline{A}(t)\underline{y}(t) + \underline{g}(t), \quad \underline{B}_0\underline{y}(0) + \underline{B}_1\underline{y}(1) = \underline{e} \quad .$$

Writing $h_i = t_{i+1} - t_i$, $\underline{A}_i = \frac{1}{2}\underline{A}(t_i + h_i/2)$, and $\underline{g}_i = \underline{g}(t_i + h_i/2)$, the equations (3.2), (3.3) take the form

$$(\underline{I}-h_i\underline{A}_i)\underline{Z}_{i+1} = (\underline{I}+h_i\underline{A}_i)\underline{Z}_i + h_i\underline{g}_i \quad \text{for } 1 \le i \le N-1$$

$$\underline{B}_0\underline{Z}_1 + \underline{B}_1\underline{Z}_N = \underline{e}.$$

In block matrix notation, this is

$$(3.6) \quad \begin{bmatrix} \underline{B}_0 & \underline{0} & \underline{0} & \cdots & \underline{B}_1 \\ -\underline{P}_1 & \underline{M}_1 & \underline{0} & \cdots & \underline{0} \\ \underline{0} & -\underline{P}_2 & \underline{M}_2 & \underline{0} & \cdots \\ & & & & \\ \underline{0} & \cdots & \underline{0} & -\underline{P}_{N-1} & \underline{M}_{N-1} \end{bmatrix} \begin{bmatrix} \underline{Z}_1 \\ \underline{Z}_2 \\ \underline{Z}_3 \\ \vdots \\ \underline{Z}_N \end{bmatrix} = \begin{bmatrix} \underline{e} \\ h_1\underline{g}_1 \\ h_2\underline{g}_2 \\ \\ h_{N-1}\underline{g}_{N-1} \end{bmatrix}$$

involving an almost (except for the first block row) bi-diagonal matrix, where $\underline{\underline{P}}_i = \underline{\underline{I}} + h_i \underline{\underline{A}}_i$, $\underline{\underline{M}}_i = \underline{\underline{I}} - h_i \underline{\underline{A}}_i$. Although $N$ may be quite large in order to obtain much accuracy, the special structure of the $Nn \times Nn$ matrix in (3.6) allows the system to be solved efficiently.

## Finite Differences for Variational Problems

We saw in Subsection 2.2 that second-order problems (1.2) subject to separated linear boundary conditions (2.2) and not involving $y'$ explicity in the differential equation are often equivalent to minimizing $J[y]$ in (2.1). Since $J$ involves an integral and a derivative, the natural finite difference approach is to use a quadrature sum and a divided difference. A simple example is to let the $n \times 1$ vector $\underline{Z}_i$ approximate $\underline{y}(t_i)$ and to let the vectors $\underline{Z}_i$ be chosen to minimize

$$J_n[\underline{Z}] = \sum_{i=1}^{N-1} h_i \{ \frac{1}{2} [(Z_{i+1} - Z_i)/(t_{i+1} - t_i)]^2 + F(t_i, Z_i) \} \tag{3.7}$$

subject to

$$Z_1 = a, \; Z_N = b .$$

More complicated differences or quadratures lead to more involved functions $[J_n]$ to be minimized.

## Finite Differences for Shooting and its Variants

All of the variants of shooting described in Subsection 2.3 involve solving some initial-value problems for ordinary differential equations as an intermediate step on the way to solving a set of algebraic equations. Finite differences can come into play be providing a way to solve these

initial-value problems approximately. Any of the high quality initial-value codes such as those for variable-order Adams methods or Runge-Kutta-Fehlberg methods can be used to solve the initial-value problem. As a trivial example for multiple shooting we can replace $\underline{y}_i' = \underline{f}(t,\underline{y}_i(t))$ for $T_i < t < T_{i+1}$ and $1 \le i \le k-1$ by

$$(\underline{Z}_{i,j+1} - \underline{Z}_{i,j})/(t_{i,j+1} - t_{i,j}) = \underline{f}(\tfrac{1}{2}t_{i,j+1} + \tfrac{1}{2}\underline{Z}_{i,j+1} + \tfrac{1}{2}\underline{Z}_{i,j}) \tag{3.8}$$

where $\underline{Z}_{i,j}$ approximates $\underline{y}(t_{i,j})$ and

$$T_i = t_{i,1} < t_{i,2} < \cdots < t_{i,N_i} = T_{i+1} ,$$

and we can replace the initial condition $\underline{y}_i(T_i;\underline{z}_i) = \underline{z}_i$ with

$$\underline{Z}_{i,1} = \underline{z}_i \cdot \tag{3.9}$$

Given the initial data $\underline{z}_i$ for multiple shooting we can use (3.8) to evaluate successively $\underline{Z}_{i,2},\underline{Z}_{i,3},\cdots,\underline{Z}_{i,N_i}$ from $\underline{Z}_{i,1} = \underline{z}_i$. For example, in the linear case in which the original equation is

$$\underline{y}'(t) - \underline{A}(t)\underline{y}(t) + \underline{g}(t) \quad \text{for } 0 < t < 1, \ \underline{B}_0\underline{y}(0) + \underline{B}_1\underline{y}(1) = \underline{e}$$

and in (2.4), the recursion (3.8) becomes merely

$$\underline{M}_{i,j}\underline{Z}_{i,j+1} - \underline{P}_{i,j}\underline{Z}_{i,j} = h_{i,j}\underline{g}_{i,j} \tag{3.10}$$

where

$$\underline{M}_{i,j} = \underline{I} - \tfrac{1}{2}h_{i,j}\underline{A}(t_{i,j} + \tfrac{1}{2}h_{i,j}), \ \underline{P}_{i,j} = \underline{I} + \tfrac{1}{2}h_{i,j}\underline{A}(t_{i,j} + \tfrac{1}{2}h_{i,j}),$$

$$\underline{g}_{i,j} = \underline{g}(t_{i,j} + \tfrac{1}{2}h_{i,j}), \text{ and } h_{i,j} = t_{i,j+1} - t_{i,j} \cdot$$

Because we are using shooting we must also satisfy the continuity conditions

$$\underset{\sim}{Z}_{i,N_i} = \underset{\sim}{Z}_{i+1,1} \tag{3.11}$$

and finally the boundary conditions

$$\underset{=0}{B}\underset{\sim}{Z}_{1,1} + \underset{=1}{B}\underset{\sim}{Z}_{k,N_k} = \underset{\sim}{e} \tag{3.12}$$

## Finite Differences for Quasilinearization

Quasilinearization discussed in Subsection 2.4 merely transformed the original nonlinear problem to a sequence of linear boundary-value problems. We can therefore use finite differences, for example as in (3.6), to solve each of these linear problems. In many cases using finite differences on the linearized differential equation as described here is the same as linearizing the nonlinear equations that result from applying finite differences to the nonlinear differential equation.

## Finite Differences for Continuation and Embedding

Finite differences can be used on the continuation problem just as it was on the original (untransformed) problem. On the other hand, for the embedding methods we needed to solve initial-value problems like (2.14), (2.15), or (2.18). High quality finite difference methods for initial-value problems can therefore be applied to solve these just as for shooting methods.

Finite Differences for Integral Equations

We saw in Subsection 2.6 how we could transform, for example, (2.19)-(2.10) with $f$ independent of $y'$ into the integral equation (2.22), namely

$$y(t) = \int_0^1 G(t,\tau)f(\tau,y(\tau))d\tau$$

Letting $t_1 = 0 < t_2 < \cdots < t_N = 1$ and approximating $y(t_i)$ by $Z_i$ as usual, we can replace the integral in the equation with a quadrature sum. Using the simple rectangle rule, for example, yields

$$Z_i = \sum_{j=1}^{N-1} (t_{j+1} - t_j)G(t_i,t_j)f(t_j,Z_j) \quad . \tag{3.13}$$

As usual, more complicated quadrature formulas yield solutions of higher order accuracy.

3.2 Projections. As we saw in Subsection 3.1, various discrete problems result from using finite differences on the various Transformed Problems. Although we considered six different types of Transformed Problems, there were only four different ways finite differences were used: i) directly on boundary-value problems (in the original problem, in quasi-linearized problems, and in continuation problems); ii) on variational problems; iii) on initial-value problems (in shooting and its variants, and in embedding methods); and iv) on integral equations. We will see in this Subsection that the projection approach to discrete modeling gives different models in the same four ways. First we sketch the projection idea itself.

The projection approach can be viewed as a way to approximate the solution to an equation

(3.14) $\quad Dx = F(x)$

where $D$ is a linear operator from some linear vector space $X$ into some linear vector space $Y$ and $F$ is a possibly nonlinear operator from $X$ into $Y$. We choose $X_h$ to be some finite dimensional subspace of $X$ and we let $Y_h$ be the finite-dimensional subspace of $Y$ defined by $Y_h = DX_h$. Finally let $P_h$ be some linear projection of $Y$ into $Y_h$, so that $P_h$ is a linear operator for which $P_h y_h = y_h$ for all $y_h$ in $Y_h$. The main idea is to seek an approximate solution $x_h$ to (3.14) in $X_h$ rather than in $X$; if $x_h$ is in $X_h$ however then $Dx_h$ is in $Y_h$ while generally $F(x_h)$ is not in $Y_h$ so that (3.14) cannot be solved in $X_h$. Instead we modify $F(x_h)$ to $P_h F(x_h)$ to get it into $Y_h$. Thus we approximate $x$ solving (3.14) by $x_h$ solving

(3.15) $\quad Dx_h = P_h F(x_h)$ ,

a finite-dimensional problem. General theorems are known on the existence of $x_h$ and on the error $x - x_h$; these theorems involve the approximation properties of the subspaces $X_h$ and $Y_h$ and the precise nature of the projection $P_h$.

For our problems, $X$ and $Y$ are spaces of functions defined on $0 \le t \le 1$, and elements $y_h$ of $Y_h$ are usually represented as linear combinations

$$y_h = a_1\theta_1 + a_2\theta_2 + \cdots + a_L\theta_L$$

of simple basis functions $\theta_1, \ldots, \theta_L$ for $Y_h$ of dimension $L$, so that (3.15) defines a set of equations in the unknows $a_1, \ldots, a_L$.

Important choices for the subspaces $X_h$ and $Y_h$ are the spline spaces $S(\Pi,k,r)$. Here $\Pi$ is a set of break points $\xi_0 = 0 < \xi_1 < \xi_2 < \cdots < \xi_\ell = 0$. and $k$ and $r$ are integers with $k \geq 0$, $r \geq -1$. An element of $S(\Pi,k,r)$ is a $r$ times continuously differentiable (often vector-valued) function on $0 \leq t \leq 1$ which is defined by a (different) polynomial of degree at most $k-1$ on each interval $\xi_i < t < \xi_{i+1}$. A sufficiently smooth function $f$ on $0 \leq t \leq 1$ can usually be approximated by some spline $\sigma$ in $S(\Pi,k,r)$ to order $k$ in the sense that $\max_{0 \leq t \leq 1} |f(t) - \sigma(t)| = \mathcal{O}(|\Pi|^k)$ where $|\Pi| = \max_{0 \leq i \leq \ell-1} |\xi_{i+1} - \xi_i|$. An important fact is that every element of $S(\Pi,k,r)$ can be written as a linear combination of special basis splines (B-splines) each of which vanishes identically on all but a few adjacent intervals $\xi_\ell < t < \xi_{i+1}$. Such a basis is called a local B-spline basis. In practice the projection $P_h$ into $Y_h$ is usually defined by i) collocation conditions, ii) orthogonality (or Galerkin) conditions, or iii) a mixture of collocation and orthogonality conditions. Collocation conditions are of the form $(P_h y)(\eta) = y(\eta)$ for certain values $\eta$; orthogonality (or Galerkin) conditions are of the form $\langle P_h y - y, \Psi \rangle = 0$ where $\Psi$ is some function and $\langle \cdot, \cdot \rangle$ denotes some inner product (usually involving integrals) on our function spaces. We proceed now to discribe briefly some projection methods.

### Projection for the Original Problem, for Quasilinearization, and for Continuation

As we saw in Subsection 3.1, the Tranformed Problems in these three cases all are still explicitly described as two point boundary-value

problems; to be specific, consider

$$\underline{y}'(t) = \underline{f}(t,\underline{y}(t)) \quad \text{for } 0 < t < 1, \ \underline{B}_0\underline{y}(0) + \underline{B}_1\underline{y}(0) = \underline{0} \ ,$$

where we assume homogeneous boundary conditions for convenience. Here we can think of the space $X$ as, say $C^1[0,1]$, $Y$ as $C[0,1]$, the operator $D$ as $\frac{d}{dt}$, and the operator $F$ as taking $\underline{y}$ into $\underline{f}(t,\underline{y}(t))$. If we take $X_h$ to be the spline subspace $S(\Pi,k,r)$ subject to our homogeneous boundary conditions above, then $Y_h = DX_h$ is essentially $S(\Pi,k-1,r-1)$ (subject to some boundary conditions). Therefore if $P_h$ denotes some projection into this modified $S(\Pi,k-1,r-1)$, our projection reduces to finding $\underline{\sigma}$ in $S(\Pi,k,r)$ *satisfying*

$$\underline{\sigma}' = P_h\underline{f}(t,\underline{\sigma}), \ \underline{B}_0\underline{\sigma}(0) + \underline{B}_1\underline{\sigma}(1) = \underline{0} \ .$$

If, for example, $P_h$ is defined by some collocation conditions $(P_h y)(\eta) = y(\eta)$, then we impose

$$\underline{\sigma}'(\eta) = \underline{f}(\eta,\underline{\sigma}(\eta)); \tag{3.15}$$

similarly an orthogonality condition might take the form

$$\int_0^1 [\underline{\sigma}'(t) - \underline{f}(t,\underline{\sigma}(t))]\underline{\Psi}(t)dt = \underline{0} \ . \tag{3.16}$$

Note that if $\underline{\sigma}$ is expressed as a linear combination of local basis elements (B-splines),

$$\underline{\sigma} = a_1\underline{\theta}_1 + \cdots + a_L\underline{\theta}_L \ ,$$

then (3.15) for example becomes

$$(3.17) \qquad a_1 \underline{\phi}_1'(\eta) + \cdots + a_L \underline{\phi}_L'(\eta) = \underline{f}(\eta, a_1 \underline{\phi}_1(\eta) + \cdots + a_L \underline{\phi}_L(\eta)) \ .$$

Since $\underline{\phi}_i'(\eta) = \underline{\phi}_i(\eta) = 0$ except for only a few subscripts $i$ because the $\underline{\phi}$'s form a local basis, only a very few of the coefficients $a_i$ are explicitly involved in each collocation equation. This implies that each equation in the system (3.15) for $a_1,\ldots,a_L$ in fact only involves a few $a_i$; this sparsity is what makes local bases important. Approximating $\underline{y}$ by elements of $S(\Pi,k,r)$, if $P_h$ is chosen carefully by carefully selecting collocation points $\eta$ or orthogonality functions $\underline{\psi}$, gives errors $\underline{y}-\underline{\sigma}$ of optimal order $k$ for $0 \le t \le 1$. In addition there are usually special points in each interval $\xi_i \le t \le \xi_{i+1}$ of the grid $\Pi$ at which much higher accuracy is obtained [Dupont (1976)]; this superconvergence can be used to generate global approximations of this higher order, so such results are important.

### Projection for Shooting and Embedding

The important feature of the Transformed Problems produced by shooting or embedding is that they are initial-value problems. The only effect this has on the discussion just completed is to change the side conditions on the splines to initial rather than boundary conditions. In other regards, collocation and Galerkin projection methods look the same here as for problems with explicit boundary conditions. Thus we consider these no further.

### Projection for Integral Equations

If the Green's function is used to transform our original problem, say of second-order, we end up with an integral equation

$$y(t) = \int_0^1 G(t,\tau)f(\tau,y(\tau))d\tau.$$

In this case we can take $X$ and $Y$ to be $C[0,1]$, $D = I$, and $F$ as the mapping from $y$ to $\int_0^1 G(t,\tau)f(\tau,y(\tau))d\tau$. Using splines $\sigma$ to approximate $y$ and collocation, for example, to define our projection gives us conditions like

$$\sigma(\eta) = \int_0^1 G(\eta,\tau)f(\tau,\sigma(\tau))d\tau.$$

Note in this case that even if we represent $\sigma$ in terms of local basis functions $\theta_i$ the resulting problem is <u>not</u> sparse because of the terms $\int_0^1 G(\eta,\tau)f(\tau,a_1\theta_1(\tau) + \cdots + a_L\theta_L(\tau))d\tau$ which have contributions from every $a_i$.

## Projection for Variational Problems

Although projection as I have described it does not strictly apply to variational problems, the spirit of projection does apply. Recall from Subsection 2.2 that we are considering the problem of minimizing

$$J[y] = \int_0^1 \{ \tfrac{1}{2}(y'(t))^2 + F(t,y(t))\}dt$$

subject to

$$y(0) = 0,\ y(1) = 0\ .$$

One of the ideas behind projection was to seek an approximate solution in a finite-dimensional subspace. Applying the same idea here we replace $y$ by $a_1\theta_1 + \cdots + a_L\theta_L$ for some chosen functions $\theta_1,\ldots,\theta_L$ and then choose $a_1,\ldots,a_L$ to minimize

$$\tilde{J}(a_1,\ldots,a_L) = J[a_1\sigma_1 + \cdots + a_L\sigma_L] .$$

This is commonly called the Rayleigh-Ritz method. It is in fact strongly related to a projection method since extremizing $J$ is strongly related to making $\nabla\tilde{J} = \underline{0}$; this is the same as using projection with orthogonality conditions determined by $\sigma_i$ and the inner product $\langle \sigma_i, g \rangle = \int_0^1 \sigma_i(t)g(t)dt$.

## 4. SOLUTION TECHNIOUES FOR DISCRETE MODELS

This Section contains much less detail than its predecessors. Primarily I want to emphasize the fact that transforming a problem (as in Section 2) and then developing a finite-dimensional discrete model of the Transformed Problem (as in Section 3) do not a method make! There still remains the formidable task of solving for the solution of the discrete problem, and there are usually very many computational techniques for doing this; only when the solution technique is known is the complete algorithm for the boundary-value problem finally specified. Each of our discrete models produced either a finite-dimensional minimization problem, or a finite system of nonlinear algebraic equations, or a finite system of linear algebraic equations.

Quite a number of distinct methods are available for minimizing a function of several variables [Murray (1972)], the problem which results from discrete models of the variational version of boundary-value problems; conjugate direction and variable metric methods are among the most powerful. Because our problems often have special structure, such as having many variables and a sparse Hessian, techniques should be used which are designed to make use of such structure.

Similarly many methods are available [Ortega-Rheinboldt (1970)] for solving the finite systems of nonlinear algebraic equations which result from the original problem, the shooting, and the integral equation approaches to boundary-value problems; among the most popular are the quasi-Newton update methods which essentially use Newton's method only with rough approximations to the Jacobian matrix of the nonlinear system. The systems arising from the original problem and from the integral equation approaches usually have many more unknowns than in the shooting systems. Systems for the original problem are usually sparse while those for the integral equation are usually dense. Again special methods should be used depending on the system's structure.

Since systems of linear algebraic equations often arise from methods to solve nonlinear equations as well as from linear differential equations with linear boundary conditions, methods for solving linear algebraic systems are fundamental to solution techniques for our discrete models. Again, although we often think only of straightforward Gauss elimination, there are many techniques available for solving linear systems. A wide variety of iterative methods can be used, especially for sparse problems. In addition, there are many choices as to how to perform direct elimination. Consider, for example, finite differences for the original problem which leads to a linear system as in (3.6) that is almost block bi-diagonal. Different procedures result from considering the matrix as full, as banded, as block banded, et cetera; these differ primarily in how much information is retained about the location of zeroes. The work required and the accuracy obtained will also differ among the procedures, and it is important to determine which procedures are "best" for solving a given discrete problem.

One of the important practical areas of investigation now is that of determining which solution technique should be used on a given discrete model. This is a vital area since the efficiency and accuracy of the solution technique can make or break a complete algorithm; I have heard of a high-quality multiple-shooting code which was improved by a factor of ten by improving its nonlinear equation solver.

## 5. RELATIONSHIPS AMONG COMPLETE ALGORITHMS

We have indicated that a complete algorithm is specified by three "co-ordinates": a Transformed Problem, a Discrete Model, and a Solution Technique. Unfortunately, different sets of co-ordinates can describe identical (or very similar) complete algorithms. In this Section I want to indicate a few relationships among such algorithms. I will not compare algorithms in the sense of saying which is "better", since that question can only be addressed in terms of specific computer codes implementing the algorithms, specific sets of test-problem classes, and specific criteria for measuring "goodness"; there is a great need for such comparisons to be performed with the same care that went into the testing of codes for initial-value problems [Hull (1975), Hull et al. (1972), Davenport et al. (1975)].

We remarked earlier on one equivalence among algorithms. In using the Rayleigh-Ritz discrete model for the variational formulation of the problem, if we set equal to zero the gradient of the function of finitely many variables to be minimized we obtain a Galerkin projection model for the original boundary-value problem. Similarly if we set to zero the gradient of the function to be minimized in the finite-difference method

of the variational formulation, then we obtain a finite-difference model of the original problem.

It has also been shown that certain spline collocation procedures are identical to finite-difference procedures when both are applied to the original problem. For example, using the spline space $S(\Pi,2,0)$ of continuous piecewise linear functions on the first-order problem (1.1) with collocation at the middle of interval between break points yields precisely the finite-difference equations (3.2) if the points $t_i$ are the break-points and $\underset{=}{Z}_i$ denotes the value of the spline at $t_i$.

At present it seems to be generally believed that the most competitive methods for boundary-value problems are based on projection for the original problem, finite differences for the original problem, finite differences for shooting and its variants, and finite differences for embedding; I therefore want to look briefly at the relationships among these procedures. We have already seen a relationship between finite differences and projection for the original problem, so I want to examine finite differences for the original problem, for shooting, and for embedding. For linear problems the relationships are very striking, since the overall methods can be identical! For nonlinear problems the same methods are not necessarily identical but are very similar. Simply to convey the idea here we look at linear first-order problems.

Consider first simple shooting for the linear scalar problem

$$y' = A(t)y + g(t),\ B_0y(0) + B_1y(1) = e$$

where we implement shooting by the simple finite-difference method (3.2); letting $Z_i$ approximate $y(t_i)$, we get the recursion formulas appearing just before (3.6). Suppose, to be precise, we take six points

$t_0 = 0 < t_1 < t_2 < t_3 < t_4 < t_5 < t_6 = 1$. Then for simple shooting we try to find $z$ so that choosing

$$Z_1 = z,\ M_i Z_{i+1} = P_i Z_i + h_i g_i \quad \text{for } 1 \le i \le 5,\ B_0 z + B_1 Z_6 = e$$

In shooting we solve the above recursion for $Z_6$ in terms of $z$ and then use this plus the boundary condition $B_0 z + B_1 Z_6 = e$ to select $z$ correctly. If we write the above recursion and boundary condition in matrix notation, we obtain

$$(5.1)\quad \begin{bmatrix} B_0 & 0 & 0 & 0 & 0 & B_1 \\ -P_1 & M_1 & 0 & 0 & 0 & 0 \\ 0 & -P_2 & M_2 & 0 & 0 & 0 \\ 0 & 0 & -P_3 & M_3 & 0 & 0 \\ 0 & 0 & 0 & -P_4 & M_4 & 0 \\ 0 & 0 & 0 & 0 & -P_5 & M_5 \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \\ Z_5 \\ Z_6 \end{bmatrix} = \begin{bmatrix} e \\ h_1 g_1 \\ h_2 g_2 \\ h_3 g_3 \\ h_4 h_4 \\ h_5 g_5 \end{bmatrix}$$

which is precisely (3.6), the equations we solved for finite differences applied to the original problem. Thus finite differences for simple shooting and for the original problem give the same answers. Moreover, we can interpret the way in which shooting solves for $Z_6$ in terms of $z = Z_1$ in the language of an elimination method for solving (3.6) or (5.1) for finite differences on the original problem. In (5.1), use the (2,2) element $M_1$ to eliminate the (3,2) element $-P_2$ and divide the second row by the (2,2) element. Next use the (3,3) element to eliminate the (4,3)

element and divide row three by the (3,3) element. Keeping this up we eventually transform the matrix in (5.1) into

$$
(5.2) \qquad \begin{bmatrix} B_1 & 0 & 0 & 0 & 0 & B_1 \\ X & I & 0 & 0 & 0 & 0 \\ X & 0 & I & 0 & 0 & 0 \\ X & 0 & 0 & I & 0 & 0 \\ X & 0 & 0 & 0 & I & 0 \\ X & 0 & 0 & 0 & 0 & I \end{bmatrix}
$$

where X denotes the presence of some nonzero element. The last row of (5.2) expresses $Z_6$ in terms of $Z_1$, as in shooting. If we now solve for $Z_1$ between the first and last rows in (5.2) and substitute the computed $Z_1$ into the equations (5.2), we obtain all the values $Z_i$, precisely as in shooting. Therefore, not only do we produce the same solutions by the two procedures, but also finite differences for simple shooting on linear problems is computationally step by step equivalent with a particular elimination method for solving the linear system resulting from the same finite difference method for the original problem. It is important to note above that we had to use the same finite difference grid points $t_i$ for both methods. For the boundary-value approach, these points must be chosen in advance; in shooting, initial-value codes usually select the grid points automatically. Thus the two methods are identical if we can somehow determine the appropriate grid points in the shooting approach. This illustrates the importance of selecting good grid points for finite differences applied to the boundary-value problems.

By a slight generalization on the preceding argument we can show that finite differences for multiple shooting is identical step by step with an elimination method for the finite-difference method for the boundary-value problem. Likewise, both simple and multiple superposition are identical with an elimination process. Somewhat more complex is the fact that the re-orthogonalization process implemented with finite differences is identical step by step with a solution technique for the finite difference equations (3.6) for the original problem; this solution technique first eliminates as for multiple shooting and then performs an orthogonal simularity transformation (based on the matrices $Q_i$ in (2.8)) so as to make all of the entries in the matrix upper-triangular as in (2.8). Also it can be shown that finite differences on the sweep method of embedding is identical with standard Gauss elimination in (3.6).

Thus finite differences for the original problem, for shooting and its variants and for embedding only differ by being different solution techniques for the same set of equations (3.6). This does not mean that the methods are not very different; different solution techniques can have drastically different results in the presence of rounding error. What our statement does mean is that it is reasonable to concentrate on alternatives to Gauss elimination in order to solve (3.6). Again I observe also that finite-difference methods for shooting and embedding have the ability to select grid points dynamically, while the finite-difference method for the original problem selects grid points in advance.

## 6. ACCELERATING CONVERGENCE AND ESTIMATING ERRORS

For simplicity we begin the discussion of this topic in a simpler

setting than differential equations. Suppose that there is some marvelous number $Y_0$ that we wish to compute, and that as some positive parameter $h$ tends to zero we are instead able to compute some approximation $Y(h)$ to $Y_0$. The problem of <u>error estimation</u> is obviously that of estimating the size of the error $Y(h) - Y_0$; the problem of <u>acceleration</u> is that of generating another scheme $\tilde{Y}(h)$ for which its error $\tilde{Y}(h)-Y_0$ is "much" smaller than $Y(h)-Y_0$. The two problems are closely related: if $e(h)$ is an accurate estimate of $Y(h)-Y_0$ then surely $\tilde{Y}(h) \equiv Y(h)-e(h)$ is an accelerated estimate of $Y_0 = Y(h) - (Y(h)-Y_0)$, while if some $\tilde{Y}(h)$ is "much" nearer $Y_0$ than is $Y(h)$ then surely $e(h) \equiv Y(h)-\tilde{Y}(h)$ is very near $Y(h)-Y_0$ which is the true error. I will first phrase my discussion in terms of accelerating convergence.

Three convergence-acceleration devices which have been used for boundary-value problems are Richardson extrapolation [Joyce (1971)], iterated deferred correction [Pereyra (1967)], and iterated defect correction [Frank (1976)]. Richardson extrapolation computes both $Y(h)$ and $Y(rh)$ for some $r < 1$ and uses some theoretical information on the behavior of $Y(h)-Y_0$ in order to compute an improved $\tilde{Y}(h)$; for differential equations this involves computations on two discretizations in order to improve the accuracy on the less accurate of the two solutions (the one with the more crude discretization). Both deferred correction and defect correction are much more complicated than Richardson extrapolation but have the advantage of avoiding computations on a refined discretization. Experiments indicate the deferred correction and defect correction are more efficient than Richardson extrapolation, but the

methods are too complex to explain here.

There are a number of other methods for estimating errors $Y(h)-Y_0$, most of which require knowledge of the asymptotic nature of the error. For example, if it is known that

$$Y(h)-Y_0 = T(Y)h^p + \mathcal{O}(h^{p+1})$$

for known $p > 0$ and for a known expression $T(Y)$ in $Y$, then one can first compute $Y(h)$ and then estimate the error by $T_h(Y(h))h^p$ where $T_h(y)$ denotes some approximation scheme for estimating $T(Y)$, such as a divided difference to approximate a derivative.

Returning to the discussion of differential equations, what we really want is an estimate of the error in approximaing a solution $y(t)$ at each $t$; several of the estimation schemes suggested above have been used to do just that for discrete models of the original boundary-value problem. In shooting and embedding we are solving initial-value problems, and most codes for such problems estimate the <u>local</u> or <u>one-step</u> error rather than the global or total error we desire. Clearly the two errors are related, but it is not fair to say, as some have, that error estimation is easier for initial-value problems; the fallacy of the statement comes from measuring two different errors.

## 7. ERROR CONTROL AND PARAMETER SELECTION

There would be little point to Section 6 on estimating error if we had no use for the estimate. What we usually want is to <u>control</u> the error, in the sense that we want to make the error less than some tolerance provided by a user of our method at as low a cost as possible; usually this means controlling the error so that it will be only a little smaller

than the tolerance. In our little mythical example in Section 6 of computing something called $Y(h)$ to approximate $Y_0$, once $h$ is chosen the error is determined; thus to control the error in any way we must appropriately select the value of our parameter $h$. In the real methods we discussed in Sections 2,3, and 4, there are many parameters at our disposal. The first, of course, are the Transformed Problem, the Discrete Model, and the Solution Technique we choose to use. After choosing these, we still face many parameters. For example: with finite differences, how many points $t_i$ to use and where to place them, and what difference approximations to use; for spline collocation, how many break points to use and where to place them, where to place collocation points, what degree and how smooth splines to use; for shooting, how many shooting points to use and where to place them; et cetera. Some methods appear to require a uniform spacing of some mesh (break points or collocation points or finite difference points); this can be disastrous on problems whose solutions change slowly in some region and very rapidly in others. Selecting the mesh in this case is very difficult; an interesting idea [Russell-Christiansen (1978)] is to use different uniform meshes on various subintervals of $0 < t < 1$ and to stop computing in a region of the interval in which the tolerance has been met. While the user of a computer code can sometimes wisely select parameters, in many cases a good choice of parameters depends on properties of the solution about which the user has no ideas. For this reason an important trend in code development is the inclusion of procedures which automatically select parameters in an attempt to attain the desired error efficiently. This is a vital research problem on which some progress is being made but where much remains to be done.

## 8. CONCLUSIONS

My aim in this paper has been to explain briefly what each of a variety of methods is, how methods relate to one another, and where are the difficulties today that stimulate interesting research problems. To describe what the methods are and how they relate we viewed each method as a Solution Technique for some Discrete Model of a Transformed Problem; this was done in the setting of two simple model problems (1.1), (1.2) but extends readily to most other types of boundary-value problems involving eigenvalues, multi-point boundary conditions, et cetera. Those areas which in my opinion deserve much more study and development include: numerical effects and efficiency of different methods of solving the linear algebraic systems that arise; methods for solving the special nonlinear algebraic systems that arise; comparative performance of codes implementing various methods on carefully chosen classes of test problems; and methods for estimating and controlling global errors by automatic selection of parameters of the method.

## 9. REFERENCES

1. Aktas, Z., and H.J. Stetter (1977), "A classification and survey of numerical methods for boundary-value problems in ordering differential equations," Int. J. Num. Meth. Eng., vol. 11, 771-796.

2. Aziz, A.K. (ed.)(1975), Numerical solutions of boundary-value problems in ordinary differential equations, Academic Press, New York.

3. Bellman, R.E., and R.E. Kalaba (1965), Quasilinearization and nonlinear boundary-value problems, American Elsevier, New York

4. Courant, R., and D. Hilbert (1953), Methods of mathematical physics, vol. I, Interscience, New York.

5. Davenport, S.M., L.F. Shampine, and H.A. Watts (1975), "Comparison of some codes for the initial-value problem for ordinary differential equations," in [Aziz (1975)], 349-354.

6. Dupont, T. (1976), "A unified theory of superconvergence for Galerkin methods for two-point boundary-value problems," SIAM J. Numer. Anal., vol. 13, 362-368.

7. Frank, R. (1976), "The method of iterated defect correction and its application to two-point boundary-value problems, I," Numer. Math., vol. 25, 409-419.

8. Hull, T.E. (1975), "Numerical solutions of initial-value problems for ordinary differential equations," in [Aziz (1975)], 3-26.

9. Hull, T.E., W.H. Enright, B.M. Fellen, and A.E. Sedgwick (1972), "Comparing numerical methods for ordinary differential equations," SIAM J. Numer. Anal., vol. 9, 603-637.

10. Joyce, C.C. (1971), "Survey of extrapolation processes in numerical analysis," SIAM Rev., vol. 13, 435-490.

11. Keller, H.B. (1968), Numerical methods for two-point boundary-value problems, Blaisdell, Waltham, Mass.

12. Keller, H.B. (1975), "Numerical methods for boundary-value problems in ordinary differential equations: survey and some recent results on difference methods," in [Aziz (1975)], 27-88.

13. Lynch, R.E., and J.R. Rice (1977), "A higher order difference method for differential equations," CSD-TR 244, Math. Sci., Purdue Univ., West Lafayette, Indiana.

14. Murray, W. (ed.)(1972), Numerical methods for unconstrained optimization problems, Academic Press, London.

15. Ortega, J.M., and W.C. Rheinboldt (1970), Iterative solution of nonlinear equations in several variables, Academic Press, New York.

16. Pereyra, V. (1967), "Iterated deferred corrections for nonlinear boundary-value problems," Numer. Math., vol. 10, 316-323.

17. Russell, R.D., and J.D. Christiansen (1978), "Adaptive mesh selection strategies for solving boundary-value problems," to appear in SIAM J. Numer. Anal.

18. Scott, M.R. (1973), Invariant imbedding and its applications to ordinary differential equations, Addison-Wesley, Reading, Mass.

19. Scott, M.R. (1975), "On the conversion of boundary-value problems into stable initial-value problems via several invariant imbedding algorithms," in [Aziz (1975)], 89-148.

20. Scott, M.R., and H.A. Watts (1977), "Computational solution of linear two-point boundary-value problems via orthonormalization," SIAM J. Numer. Anal., vol. 14, 40-70.

Unclassified

Security Classification

## DOCUMENT CONTROL DATA - R & D

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

1. ORIGINATING ACTIVITY *(Corporate author)*

University of Texas at Austin

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

(6) A Road Map of Methods for Approximating Solutions of Two-Point Boundary-Value Problems

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Center for Numerical Analysis

5. AUTHOR(S) *(First name, middle initial, last name)*

(10) James W. Daniel

6. REPORT DATE

(11) Jan 78

7a. TOTAL NO. OF PAGES

41 (12) 43 p.

7b. NO. OF REFS

20

8a. CONTRACT OR GRANT NO.

(15) N00014-76-C-0275

b. PROJECT NO.

9a. ORIGINATOR'S REPORT NUMBER(S)

(14) CNA-130

c.

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

d.

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Mathematics Branch, Office of Naval Research, Washington, D.C.

13. ABSTRACT

Numerical methods for approximating solutions of two-point boundary-value problems for ordinary differential equations are surveyed. Specific complete algorithms are classified according to how the original problem is transformed, how the transformed problem is modeled discretely, and how the discrete model is solved. Relationships among various complete algorithms are presented. Convergence acceleration, error estimation and control, and parameter selection are also discussed.

Key Words: Boundary-value, ordinary differential equations, numerical methods.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

406262

Unclassified

Security Classification

FGK